پروژه درس مقدمه‌ای بر یادگیری ماشین

نام و نام خانوادگی: امیرمهدی جعفری فشارکی

شماره دانشجویی: ۹۸۱۰۹۶۴۵

استاد درس: دکتر محمدزاده

تاریخ: ۴ بهمن ۱۴۰۰

فهرست مطالب

۱ توضیحات اولیه

کد پروژه به همراه تمام داده ها و نتایج آن در فایل ML_Project_98109645.ipynb قرار دارند. این کد از چند بخش تشکیل شده است.

- در بلوک اول، تمامی کتابخانه و توابع لازم به کد اضافه شده اند. کتابخانه های استفاده شده در این پروژه عبارتند از: Numpy ، Pandas ، Matplotlib ، Scikit learn و Scipy
- در بلوک بعدی، داده های آموزش و تست در متغیر های datai و datai_test ذخیره شده اند که در حقیقت i در اینجا نشان‌دهنده شماره شخص می‌باشد.
- بلوک سوم کد که شامل پیاده‌سازی کلاسی به نام Person است در حقیقت مهمترین بخش پروژه می‌باشد. در واقع برای هر کدام از اشخاص، یک نمونه از این کلاس می‌سازیم و تمام اعمال مختلف نظیر درست کردن ویژگی‌ها، انتخاب ویژگی‌ها با استفاده از Cross-Validation و Fisher Score ، انتخاب بهترین مدل، اندازه گیری دقت و پیشبینی کلمه برای داده تست و ... در این کلاس پیاده‌سازی شده است.
- در ادامه برای هر کدام از ۹ شخص، یک نمونه از کلاس ساخته شده و سپس بهترین مدل را برای آن پیدا و در نهایت کلمه متناظر با داده تست آن به دست آمده است.

در بخش‌های بعدی گزارش، ابتدا به پیاده‌سازی کلاس Person و سپس به نتایج به دست آمده برای هر کدام از افراد با جزئیات بیشتر پرداخته شده است.

۲ کلاس Person

این کلاس در حقیقت بخش اصلی پروژه را تشکیل داده و همان‌طور که پیشتر ذکر شد، برای هر شخص داده‌های آموزش به این کلاس داده شده و مدل مورد نظر برای آن شخص در درون این کلاس ذخیره شده. سپس به همین شکل با استفاده از یکی از توابع این کلاس، داده تست مورد نظر ورودی داده شده و برای آن، کلمات مورد نظر پیشبینی می‌شود. در ادامه توضیحات دقیق تر در رابطه با هر کدام از متدهای مورد استفاده در این کلاس داده شده است.

۱.۲ متد ___init___

این متد، در حقیقت فایل mat را که توسط تابع loadmat از کتابخانه scipy.io خوانده شده است را به عنوان ورودی می‌گیرد و سپس کل داده ها و همچنین زمان نمونه برداری را از روی آن استخراج می‌کند. این داده ها در متغیرهای data و t_sampling ذخیره می‌شوند.

۲.۲ متد find_t_stimulated

این متد، کل داده ها را به عنوان ورودی گرفته و آرایه ای از زمان‌هایی که در آن‌ها تحریک رخ داده است را به عنوان خروجی باز می‌گرداند. همچنین یک متغیر دیگر به عنوان test_data به عنوان ورودی به این تابع داده می‌شود که تعیین می‌کند که داده مورد نظر که قرار است زمان‌های تحریک روی آن به دست بیاید داده تست است یا خیر. تفاوت این دو داده این است که تمام داده های ما، در حقیقت با هر بار تحریک، به اندازه ۴ نمونه تحریک مورد نظر روشن می‌ماند و از طرفی ویژگی های این ۴ نمونه پشت سر هم به دلیل شباهت بسیار زیاد، عملا باعث می‌شود که به نوعی از هر داده در آموزش، ۴ کپی یکسان داشته باشیم. این موضوع در هنگام آموزش می‌تواند موجب به وجود آمدن overfit در داده ها و کاهش دقت در داده تست شود. به همین دلیل در داده آموزش یعنی زمانی که test_data برابر با False است، نمونه ها ۴ تا ۴ تا به خروجی داده می‌شود. به عبارت دیگر نمونه ها را با نرخ ۴ Downsample می‌کنیم.

۳.۲ متد epoching

این متد، داده ها و بردار زمان تحریک را به عنوان ورودی می‌گیرد و برای هر زمان تحریک و هر کانال، ۱۲۸ داده حول آن زمان از یک دهم قبل تا چهار دهم ثانیه بعد از زمان تحریک را ذخیره می‌کند و به عنوان یک آرایه با ابعاد تعداد نمونه در ۱۰۲۴ به عنوان خروجی باز می‌گرداند. همچنین باز هم یک متغیر test_data به عنوان ورودی داده می‌شود که صرفا در حالتی که داده آموزش باشد، علاوه بر این داده‌های زمانی، برچسب های مربوط به آن نیز به عنوان خروجی داده می‌شود.

۴.۲ متد create_all_features

این متد جزو مهمترین متدهای این کلاس است. در این متد، نمونه‌های زمانی استخراج شده توسط متد epoching به عنوان ورودی به این متد داده می‌شود و در خروجی، ویژگی‌های مربوط به آن نمونه‌های زمانی به عنوان خروجی داده می‌شوند. این ویژگی‌ها عبارتند از:

- میانگین گرفته شده ۱۲۸ داده روی ۸ کانال به عنوان یک داده ۱۲۸ تایی زمانی
- هیستوگرام برای هر ۸ کانال که توسط متد create_hist_feature ساخته می‌شود و داده ها را در بازه 60- تا 60 به ۱۲ بخش تقسیم کرده و تعداد داده ای که در هر بخش قرار می‌گیرد را به عنوان یک ویژگی می‌دهد.
- میانگین داده های هر کانال
- واریانس داده های هر کانال
- همبستگی کانال ها با یکدیگر که توسط متد create_correlation_feature ۲۸ ویژگی برای هر نمونه تولید می‌کند.
- ویژگی های فرکانسی که توسط متد create_frequency_features تولید می‌شوند و عبارتند از:
  - انرژی ۵ بازه فرکانسی نیم تا ۴ هرتز، ۴ تا ۸ هرتز، ۸ تا ۱۳ هرتز، ۱۳ تا ۳۰ هرتز و ۳۰ هرتز به بعد
  - فرکانس میانگین
  - ۵ فرکانس با بیشترین پیک

۵.۲ متد fisher_score_index

این متد در حقیقت ماتریس متشکل از ویژگی‌ها و برچسب ها را به عنوان ورودی گرفته و برای هر ستون (که نشان دهنده هر ویژگی است) fisher score را محاسبه کرده و به ترتیب از بیشترین به کمترین خروجی می‌دهد.

۶.۲ متد cross_validation

این متد، در حقیقت روی یک داده آموزش به ازای یک مدل مشخص و تعداد ویژگی مشخص (یعنی تعداد بهترین ویژگی ها بر اساس امتیاز فیشر)، cross validation انجام داده و امتیاز را برای مدل روی داده داده شده محاسبه و به عنوان خروجی می‌دهد.

۷.۲ متد scorer

این متد بر اساس متریکی که به عنوان ورودی به آن می‌دهیم، امتیاز را برای یک داده مشخص محاسبه می‌کند. همچنین لازم به ذکر است که متریک‌های قابل استفاده عبارتند از accuracy و roc_auc.

۸.۲ متد find_best_model

این متد، نمونه ها و ویژگی های مرتبط به آن را به همراه برچسب های مربوط گرفته. سپس ۲۰ درصد این داده ها را به عنوان تست و مابقی را به عنوان آموزش در نظر گرفته و از روی آن fisher score را برای هر ویژگی محاسبه و سپس روی مدل ها و تعداد ویژگی های متفاوت، cross validation انجام داده و بهترین مدل را (به همراه متغیر های دیگر نظیر اندیس های fisher score ، داده های تست و آموزش تقسیم شده و ...) به عنوان خروجی بازگردانده.

۹.۲ متد fit

این متد، ابتدا بهترین مدل را با استفاده از متد find_best_model که پیشتر معرفی شد یافته و سپس، امتیاز های متفاوت شامل accuracy و roc_auc و همچنین confusion matrix را برای بهترین مدل محاسبه و چاپ کرده. همچنین پس از آن، مدل را با تمام داده ها آموزش داده و این مدل را ذخیره کرده و برای این مدل نهایی نیز confusion matrix را محاسبه کرده.

۱۰.۲ متد load_test_data

این متد داده mat تست خوانده شده توسط تابع loadmat را به عنوان ورودی گرفته و ویژگی‌های آن، کاراکتر های تحریک، زمان تحریک و دیگر پارامترهای مرتبط با آن را استخراج و ذخیره کرده.

۱۱.۲ متد find_test_word

این متد کلمه مورد نظر شخص در داده تست را پیشبینی می‌کند. برای اینکار ابتدا تشخیص داده می‌شود که نحوه نمایش به شکل Single Character یا به شکل Row-Column می‌باشد. برای اینکار بیشترین عدد نمایش داده شده در ردیف ۱۰ ام چک شده و در صورتی که این عدد بیشتر از ۱۲ باشد، نحوه نمایش Single Character و در غیر این صورت به صورت Row-Column می‌باشد. پس از تشخیص نحوه نمایش، از یکی از دو متد display_type1 یا display_type2 استفاده شده تا کلمه مورد نظر بر اساس نحوه نمایش پیدا شود.

۱.۱۱.۲ متد display_type1

در صورتی که نحوه نمایش به صورت Single Character باشد، از این متد استفاده شده و این متد ابتدا کل زمان را به ۵ بخش تقسیم کرده و در هر بخش، حرفی که بیشترین تکرار داشته و برچسب مربوط به آن ۱ پیشبینی شده است را به عنوان یک حرف در نظر گرفته و در نهایت با چسباندن این ۵ حرف به یکدیگر، کلمه نهایی را به دست می‌آورد.

۲.۱۱.۲ متد display_type2

این متد در زمانی که نحوه نمایش به صورت Row-Column باشد استفاده شده و نحوه کار آن این است که بین حالت هایی که تحریک دارای برچسب ۱ بوده یعنی واقعا شخص در حال نگاه کردن به آن کلمه بوده، حالت‌هایی که دقیقا پس از یک ستون، یک ردیف هم دارای برچسب ۱ بوده و یا برعکس را در نظر گرفته و سپس بازه زمانی را به ۵ قسمت تقسیم و حرفی که بیشترین تکرار را داشته را به عنوان حرف مورد نظر در نظر گرفته و در نهایت با کنار هم قرار دادن این حروف، کلمه نهایی را ساخته و به عنوان خروجی باز می‌گرداند.

۳ نتایج به دست آمده برای هر شخص

۱.۳ شخص ۱

- نحوه نمایش: تک حرفی
- بهترین مدل: LDA با احتمال پیشین برابر
- دقت: 80.93%
- roc_auc: 56.47%
- ماتریس در هم ریختگی: $\begin{bmatrix} ۴۳۳ & ۹۴ \\ ۹ & ۴ \end{bmatrix}$
- کلمه نهایی: 0WBEW

۲.۳ شخص ۲

- نحوه نمایش: تک حرفی
- بهترین مدل: لاجیستیک
- دقت: 72.04%
- roc_auc: 47.47%
- ماتریس در هم ریختگی: $\begin{bmatrix} ۳۸۵ & ۱۳۶ \\ ۱۵ & ۴ \end{bmatrix}$
- کلمه نهایی: 1UKAS

۳.۳ شخص ۳

- نحوه نمایش: سطری-ستونی
- بهترین مدل: LDA با احتمال پیشین برابر
- دقت: 69.44%
- roc_auc: 65.50%
- ماتریس در هم ریختگی: $\begin{bmatrix} ۱۰۶ & ۴۲ \\ ۱۳ & ۱۹ \end{bmatrix}$
- کلمه نهایی: 7O2AD

۴.۳ شخص ۴

- نحوه نمایش: سطری-ستونی
- بهترین مدل: جنگل تصادفی با ماکسیمم عمق ۴ و ۹ نخمینگر
- دقت: 73.89%
- roc_auc: 55.18%
- ماتریس در هم ریختگی: $\begin{bmatrix} ۱۲۵ & ۲۶ \\ ۲۱ & ۸ \end{bmatrix}$
- کلمه نهایی: 52ZCD

۵.۳ شخص ۵

- نحوه نمایش: سطری-ستونی
- بهترین مدل: LDA با احتمال پیشین برابر
- دقت: 63.33%
- roc_auc: 52.15%
- ماتریس در هم ریختگی: $\begin{bmatrix} ۱۰۴ & ۴۹ \\ ۱۷ & ۱۰ \end{bmatrix}$
- کلمه نهایی: XAPZT

۶.۳ شخص ۶

- نحوه نمایش: سطری-ستونی
- بهترین مدل: LDA با احتمال پیشین برابر
- دقت: 60.00%
- roc_auc: 50.25%
- ماتریس در هم ریختگی: $\begin{bmatrix} ۹۶ & ۴۹ \\ ۲۳ & ۱۲ \end{bmatrix}$
- کلمه نهایی: MJD4R

۷.۳ شخص ۷

- نحوه نمایش: سطری-ستونی
- بهترین مدل: جنگل تصادفی با ماکسیمم عمق ۶ و ۹ نخمینگر
- دقت: 70.56%
- roc_auc: 48.04%
- ماتریس در هم ریختگی: $\begin{bmatrix} ۱۲۴ & ۱۶ \\ ۳۷ & ۳ \end{bmatrix}$
- کلمه نهایی: PXLMU

۸.۳ شخص ۸

- نحوه نمایش: سطری-ستونی
- بهترین مدل: LDA با احتمال پیشین برابر
- دقت: 67.78%
- roc_auc: 58.36%
- ماتریس در هم ریختگی: $\begin{bmatrix} ۱۰۸ & ۴۰ \\ ۱۸ & ۱۴ \end{bmatrix}$
- کلمه نهایی: 3RV4D

۹.۳ شخص ۹

- نحوه نمایش: سطری-ستونی
- بهترین مدل: LDA با احتمال پیشین برابر
- دقت: 66.11%
- roc_auc: 52.96%
- ماتریس در هم ریختگی: $\begin{bmatrix} ۱۰۸ & ۳۷ \\ ۲۴ & ۱۱ \end{bmatrix}$
- کلمه نهایی: 5PZAN